**Public Key Encryption**

**Bronson Jastrow**

**University of Wisconsin - Platteville**

**Email: jastrowb@uwplatt.edu**

# Abstract

The modern internet is a place of many private transactions over an insecure network where eavesdroppers may be present. Cryptography, and specifically public key cryptography protect the confidentiality, integrity, and authenticity of exchanges between parties over insecure networks such as the internet. Public key cryptography is a key part of hybrid cryptosystems which is the foundation for SSL/TLS and is used to secure email and banking. Public key cryptography is a means of cryptographically securing digital exchanges by using two separate but related keys: one to encrypt data (the public key) and one to decrypt (the private key). If used correctly, public key cryptography with the help of certificate authorities can notify users of man-in-the-middle attacks, prevent data from being decrypted even in the case of a lost key (known as perfect forward secrecy), verify identity, and alert users to modifications made to messages. Like many fields of cryptography, public key cryptography may be implemented in different ways, but only integer factorization will be discussed for the sake of brevity.

# Introduction

Cryptography is the study and practice of securing communications between two parties, and includes four basic goals of cryptography [7].

1. *Confidentiality*: Ensure that only the intended recipients of a message have the means to read the message.
2. *Integrity*: Ensure that the intended recipients of a message have received the message unaltered by outside parties.
3. *Authentication*: Ensure that subsequent messages from a source can be verified as originating from that source.
4. *Non-repudiation*: Ensure that messages are traceable back to the originator and only the originator so that if disputes arise, the origin of the message is undeniably from the originator.

Cryptography has been used for thousands of years to secure communications between individuals and providing confidentiality. Often, cryptography was and is used to keep communication between armed forces unintelligible to possible eavesdroppers. Early cryptography took the form of simple shifts in the alphabet, and were used by famous leaders such as Julius Caesar, who reportedly used a shift of three to communicate with his generals; meaning every “a” would become “d” and every “b” would become an “e” after encryption. This method is easily decipherable and more advanced methods were needed to keep communications secure. Cryptography became increasingly complicated during WWI and WWII where the famous Enigma cipher was used to secure Nazi communication. Eventually it was realized that an encrypted message must be able to remain secure even if the algorithm to encrypt and decrypt the message is known. The modern era of cryptography relies on public key cryptography (also known as asymmetric key cryptography) as a means of communicating securely and satisfying other goals in cryptography other than just confidentiality.

# Symmetric Key Cryptography

Symmetric key cryptography was the main method of securing messages until public key encryption was invented in the 1970's. Symmetric key encryption uses a single key to encrypt and decrypt information. The symmetric key must be shared with all parties which would like to be able to communicate together via this method. Symmetric key encryption and decryption are typically faster than public key encryption because less computation is required. The downfall of symmetric key cryptography is that the key must be shared by a secured channel or eavesdroppers may intercept the key and be able to decrypt all following communications. For this reason, symmetric keys are sometimes called pre-shared keys, since exchange of keys over the network should not occur. Symmetric keys may be transferred safely over open networks such as the internet as long as proper precautions have been taken. This will be further discussed later in the section about hybrid cryptosystems which further addresses the inability of symmetric key cryptography to satisfy all the goals of a cryptographic system. [7]

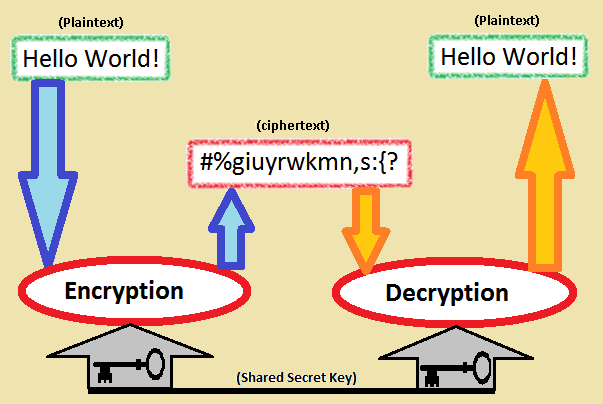


Figure 1: Symmetric Key Encryption and Decryption [3]

Figure 1 above shows a brief example of how symmetric key cryptography works. The plaintext is used in conjunction with the encryption algorithm and the symmetric key to create the ciphertext, which is the encrypted text and is unintelligible. The ciphertext is sent to the recipient, and is decrypted using the decryption algorithm and the same symmetric key which was used to encrypt it.

# Public Key Cryptography

Public key cryptography, also known as asymmetric key cryptography, is similar to symmetric key cryptography in the sense that keys are required for encryption and decryption of data. The difference between them being that public key cryptography requires two separate keys that are mathematically linked. One of these keys is called a private key, which is used to decrypt and sign messages. The private key should never be given out, and great precaution should be taken to ensure the safety of the key from being lost or stolen. The other key is called the public key which is used to verify signatures and to encrypt messages for its related private key. The public key is entirely safe to publish, as it is infeasible to determine the private key even when the public key is known. Exchanging large amounts of data with public key cryptography is slower than symmetric key cryptography, and requires larger key sizes to be considered secure compared to symmetric key sizes. [7]

Since public keys may be shared without compromising security, public keys may be exchanged between two parties over unsecured networks such as the internet without an eavesdropper being able to intercept and use the keys to decrypt messages being sent. It is however possible for the eavesdropper to intercept and replace the public key from one party with his own public key, effectively stealing the assumed identity of one party. This vulnerability will be discussed in the section about key exchange. [7]

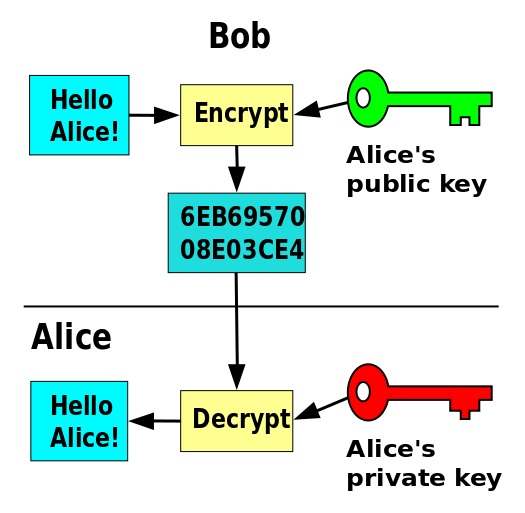


Figure 2: Public Key Encryption and Decryption [5]

Figure 2 above shows the encryption and decryption process of a message after Alice and Bob have exchanged public keys. Bob encrypts data to Alice with her public key. Alice is then able to decrypt that message using her matching private key. The original key exchange is not shown in this figure.

## Trap Door Function

Public key cryptography is implemented using trapdoor functions which are easy to compute in one direction but are difficult to compute in the opposite direction. [4] The integer factorization problem is an example of one of these trap door problems, and is used in the popular RSA public key algorithm. It is significantly more difficult to decompose a large integer into its prime factorized form than it is to multiply its prime factors into itself. Take for example the prime factorization . It is easy to compute the value of the represented integer: 864. There are currently no algorithms published that can efficiently determine the prime factorization of 864 with current computing hardware. When the prime factorization consists of two large primes of similar sizes (the product of which is called a semiprime), the problem of integer factorization becomes even more difficult. For this reason, large primes of similar size are used for public key cryptography. The largest factorization of a semiprime was done in 2009 by several research facilities on a 232 digit number in 2009. It took the equivalent of two thousand years of computing on a single core 2.2GHz AMD Opteron, illustrating the difficulty of breaking public key encryption based on integer factorization. [6]

## Digital Signatures

While not a necessity for public key encryption, digital signature algorithms are often incorporated with public key encryption software or algorithms such as the RSA algorithm. Digital signatures allow a message sender to sign the message with their private key. This signature can be verified by anyone who has access to the matching public key. A successful verification means that the message has not been tampered with, and that the sender is someone who has access to the public key matching the private key. This signature provides authentication and non-repudiation of the message, as only the holder of the private key can sign a document which will be correctly verified. Integrity is also ensured as any changes to the message since signing will result in a failure to verify the message. [7]

Generally precautions are taken to ensure that a party's private key does not fall in the hands of others, meaning that the message is from the person associated with the private key. Problems can arise if someone steals a private key from the owner. The thief can sign communications and data effectively pretending to be the private key's owner. Because of the design of private keys and digital signatures, it would be very difficult for the original key owner to deny that the thief's actions were not his own. [7] Digital signatures may also have a negative effect of removing deniability of legitimate communications. Say a journalist with a known public key signs documents with their private key and tries to send them to their publisher overseas. If that journalist is in a country where they monitor internet traffic and deny freedom of speech, it would be very difficult for the journalist to deny that they had sent the documents if the government decided to persecute them.

Hybrid Cryptosystems

Mentioned earlier in this paper was that symmetric key cryptography is faster and more efficient than public key cryptography, but lacks security when exchanging keys over unsecured channels. Hybrid cryptosystems combine the speed of symmetric key cryptography with the security of public key cryptography. A hybrid cryptosystem consists of a public key cryptosystem for key encapsulation and a symmetric key cryptosystem for data encapsulation. To use a hybrid cryptosystem, one party, party A, obtains the public key for the party they wish to send data, party B, and generate a symmetric key for the session. The data party A wishes to send to party B is encrypted using the symmetric key that they generated for the session. The symmetric key is then encrypted by party A with party B's public key, and both encrypted message and encrypted key are sent to party B. Party B can then decrypt the symmetric key using their private key, and can decrypt the data using the symmetric key they just decrypted. This system works efficiently because key sizes are relatively small (on the order of kilobytes or less) and are quickly encrypted and decrypted quickly by public key algorithms. The main data is encapsulated in symmetric key encryption which is faster with encryption and decryption than public key cryptography. [1][9]

Hybrid cryptosystems are used on a daily basis by most computer users in the form of HTTP Secure (HTTPS). HTTPS is based off of SSL/TLS (which are the specific subsystems that use hybrid cryptography) and is used to prevent eavesdropping on users connecting to secured services such as banking, email, or other online accounts. SSL/TLS incorporate certificates to verify owners certificates through the use of certificate authorities as will be described in the section on key exchange. [1]

# Key Exchange

Exchanging public keys involves a problem of trust which has not been entirely solved as of yet. When exchanging public keys, each party has to trust that the public key that they have received is owned by the person they believe to be the owner. In short, there must be a way to verify that the party sending their private key is who they say they are. The solution in place for this key exchange is a central Certificate Authority who is trusted by users and will provide digital certificates to authenticate a user's identity. [2] Well known examples of certificate authorities are VeriSign and Symantec.

When a user accesses a web page where encryption is used, such as an email login page, the email server sends a public key to the client with a certificate which contains a copy of the key signed by a certificate authority's private key. The client should already have the public key of the certificate authority either from operating system install or from the browser, and can then verify the integrity of the certificate. A valid certificate confirms the validity and authenticity of the sent public key. This prevents a man-in-the-middle attack where an attacker replaces the server's public key with their own. [1] Since an attacker cannot forge a signature from a certificate authority for their public key, their certificate would fail to be verified by the certificate authority's public key.

# Key Revocation

The idea of digital signatures works as long as long as only the owner of a private key has access to it. If the private key is stolen or copied the thief can steal the identity of the private key owner. To mitigate the damage of key loss, procedures may be put in place to handle the loss of control over a private key. A popular solution is to maintain a Certificate Revocation List (CRL) which lists an identification number for public keys that have been revoked. Keys marked as revoked should not be trusted. Reasons for key revocation is useful for use cases other than a key being compromised. Some reasons to revoke keys are: Key Compromise, Affiliation Change (employer revoking an employee's key), or Key Superseded (possibly for a stronger key) . After being revoked, the person revoking the key may list an identifier for their new key so people may still contact him. [2]

Certificate Revocation Lists must have some method of handling revocation requests. An easy solution is to allow users to revoke keys that they possess the matching private key to. This however leaves open the possibility of a key thief revoking the owner's key rendering the owner unable to let others know his new key. To counter this, CRLs can be set up to use compound principals; meaning conditions that need to be met to revoke the key. An example of a compound principal could be one where the owner and another trusted friend are given revoke authority when they both send the CRL a revocation request.. Then only the owner and the trusted friend together may revoke the key from the CRL, but neither owner nor friend could do so alone. This however creates a single point of failure where if the trusted friend is unable to, or kept from requesting revocation, then the key may not be revoked. [2]

After successful revocation, the revocation of the key must be spread to users and a new key will need to be distributed. There are several methods used depending on the needs of users. The news of revocation and issuing of a new key may be pushed from a server to clients, or downloaded by users from a centralized server. Pushing news from a server to clients is not reliable as recipients are not guaranteed to receive the updates. In fact, malicious attackers could launch a denial of service attack to intentionally keep key revocation updates from users. To counter this, a CRL may have a short expiration date so that clients must keep up to date lists of keys that have been revoked. This however means that clients must be regularly updated, and if updates are not obtained, then the client will be unable to safely conduct business. It can be seen that there are tradeoffs between uptime and security which have to be managed on a situational basis. [2]

## Perfect Forward Secrecy

What happens if a server's private key has become compromised, and an attacker has been intercepting traffic to said server for months? Unfortunately, the system described above does not stop the attacker from using the stolen private key to decrypt all communications to the server since each session key (the symmetric key) was encrypted using the user's private key. There is a solution to this problem called perfect forward secrecy. The idea is that if a private key is compromised in the future, all current and past communications will still be safe. The issue is that private keys are used for both signing identity and decrypting symmetric keys. There is an algorithm called the Diffie-Hellman key exchange which provides perfect forward secrecy of communication by creating a randomized symmetric key on both server and client end without ever exchanging the key over the network. [8] This will be explained in the next section.

## Diffie-Hellman Key Exchange

Diffie-Hellman key exchange uses the difficulty of reversing the computation of discrete logarithms to create symmetric keys on both clients without prior setup and without sending the symmetric key over an unsecured network. With a Diffie-Hellman key exchange, party *x* and party *y* decide on a prime number and a base, *p* and *b* respectively. To keep this explanation easy to read, lower case variables mean that the value is not a secret and could be publicly known, where upper case variables are secret. *p* and *b* are shared between party *x* and party *y*. Party *x* and party *y* both choose random secret integers, *SX*and *SY* respectively. Party *x* sends party *y a* where *a=*  and party *y* sends party *x c* where *c = .* The symmetric key is then computed as for party *x* and . Both of these computed values are the same, and work as a symmetric key unique to the connection. This effectively protects messages even if a private key has been compromised. Figure 3 below shows an analogy example of how the exchange would work if colors were used instead of numbers. [8] [1] [9]



Figure 3: Diffie-Hellman Key Exchange Analogy [10]

The downside with using the Diffie-Hellman key exchange is that it decreases response time between server and client. Fortunately there exists a new method of Diffie-Hellman key exchange which uses elliptic curves over finite fields. Not all browsers have support for this, and Diffie-Hellman key exchange is not used widely in practice. [8] [1] [9]

# Attacks on Public Key Systems

All public key cryptosystems depend entirely on keeping private keys in the hands of only those authorized. If an attacker gains access to a key owner's system, the attacker could gain access to the owner's private key. It is useful to keep private keys password protected so there is some time to revoke a key if the owner has reason to suspect that their key may have been compromised before an attacker breaks the password. Assuming users can successfully keep their private keys safe, there are several ways for malicious people to attack a public key system. The primary attack of a system is an attempt to get unencrypted data from encrypted data. This can be attempted on a message per message basis, or by attempting to derive the private key. If messages can be decrypted without a private key, the public key algorithm in question is considered to be broken, and if the private key can be derived, the algorithm is considered to be completely broken as this would allow the attacker easy access to all messages the private key could decrypt. [2]

Attacks on public key systems besides attacks covered already are mostly attacks on the algorithms used or implementations of those algorithms. Currently used public key systems are considered secure as long as key length is long enough to make brute-force or similar attacks (these are specialized to each algorithm) infeasible. [2]

# Conclusion

Public key cryptography provides a major portion of modern internet and data security especially when used in conjunction with traditional symmetric cryptography. Properly set up hybrid cryptosystems make up a large portion of secure networks, and appear to provide substantial security well into the future against the most determined of attackers while still providing stable and easy usability to standard users.

# References

[1] Bernat, V. (2011). *SSL/TLS & Perfect Forward Secrecy*. Retrieved 10 2, 2013, from Vincent Bernat Blog: http://vincent.bernat.im/en/blog/2011-ssl-perfect-forward-secrecy.html

[2] Cooper, A. D. (1998). *A closer Look at Revocation and Key Compromise in Public Key Infrastructures.* Gaithersburg: National Institute of Standards and Technology.

[3] Dev-NJITWILL. *Crypto.*

[4] Diffie, W., & Hellman, M. E. (1976). *New Directions in Cryptography.* IEEE.

[5] Göthberg, D. *Public Key Encryption.* Sweden, Sweden.

[6] Kleinjung, T. (2010). *Factorization of a 768-bit RSA modulus.*

[7] Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of Applied Cryptography.* CRC Press.

[8] Rescorla, E. (2006). Diffie-Hellman Key Exchange - A Non-Mathematician's Explanation. *ISSA Journal* , 7.

[9] Ristic, I. (2013, June 25). *SSL Labs: Deploying Forward Secrecy*. Retrieved 10 10, 2013, from Qualys Community: https://community.qualys.com/blogs/securitylabs/2013/06/25/ssl-labs-deploying-forward-secrecy

[10] Vinck, A. H. (2012, May 12). Introduction to Public Key Cryptography. Duisburg-Essen, Germany. Retrieved from http://www.exp-math.uni-essen.de/~vinck/crypto/script-crypto-pdf/add-to-3.pdf